Closure and stabilization in open source artefacts

Eduard Aibar and maxigas

July 4, 2014

What is closure and stabilisation?

Closure and stabilisation are two related concepts in SCOT that are used to understand the trajectory in the biography of a particular technology. Closure can be interpreted in two different ways according to whether we look at the given technology historically in its construction or we try to grasp it in its present state as a finished product. We will call these two perspectives the diachronic and synchronic view, respectively.

If we apply the diachronic view, closure marks the point where controversies end and a consensus begins to emerge about the meaning of the artefact. The initially differing visions of each relevant social group can be theorised as belonging to different technological frames, which began to overlap at this point. The capacity of a technology to absorb various interpretative frames is conceptualised as its interpretative flexibility. The interpretative flexibility of the artefact decreases through its stabilisation.

On the other hand, from the synchronic perspective, closure means that the artefact achieved a relatively stable state where substantial changes to the technology are not needed accourding to the relevant social groups, or even if needed by some of them, they are not allowed anymore. In this capacity, closure can be said to mark the end of design: when the technology is ready for the people who are aptly called its "end users". At this point reification (Lukács 1971 [1923]) is complete and people meet the given technology in their everyday life as a solid social fact – i.e. something given.

There is also a related concept in Actor Network Theory (ANT) called *black-boxing*. Since ANT conceptualises the world as a set of networks, blackboxing is understood as a stabilised network closed up to look like unified actant – that is, a node. This network-become-node is maintained by its actants in a constant configuration, but at the same time, and more crucially, its internal structure becomes mostly unproblematic, invisible and obscured (Callon 1986). The people who are aptly called its "end users" are only concerned with its specified inputs and tailored outputs (Latour 1999). In order to be able to look at the artefact as a network again, actants have to invest a considerable amount of work, which usually translates to expertise, among other things. Furthermore, in order to rearrange the network, they have to do the same to destabilise it first. Both closure and blackboxing are very visual metaphors and putting a consumer electronics device in its case when it is ready for delivery is the ultimate underlying referent in these metaphors, even if blackboxing as a concept was invented in the era of beige ABS[1] computer cases.

The virtue of closure and stabilisation as conceptual tools is that they abstract and circumscribe a certain pattern of changes in the social construction of an artefact, separate from the formal stages of the industrial design process. Given this abstraction, it is possible to see, investigate and debate where closure is located and how stabilisation takes place during the life span of a technology. Furthermore, as we will see shortly it allows us to ask how relevant social groups make closure and stabilisation the subjects of controversies, and in turn how that affects the closure and stabilisation of a piece of technology.

The weak points of closure and stabilisation as conceptual tools is that they were formulated based on empirical case studies of technologies which were always intended to be stabilised. In other words, each relevant social group around them agreed about the need to stabilise them at some point. What we say here is that there was a bias in the selection of cases. At the time closure and stabilisation were distilled into theoretical terms by STS scholars and historians of technology, which had consequences to the understanding of how we conceive of technology research and development historically. Finally these terms came bundled with the assumptions that all technology is on a journey to stabilisation — the idea of reversing stabilisation (reverse-engineering) and purposefully building unstable devices (what we call unfinished artefacts) was simply not considered. Here, we would like to further complicate the understanding of closure and stabilisation by applying them to cases where closure and stabilisation themselves become controversial. That is, it turns out that sometimes some relevant social groups approach the technologies in question as tools for opening up closures and de-stabilising technological systems. These critical and reflexive practices afford us a new understanding of the role of closure and stabilisation in the social construction of technology and their political potentials for the democratisation of design.

Closure and stabilisation in open source artefacts

It was not only the theoretical understanding of the social construction of technologies which developed over the last decades but at the same time the technology landscape also evolved. Perhaps the mass deployment of microprocessors and communication networks made their mark most. Between social theory

[1] Acrylonitrile butadiene styrene

and technological practice, engineering cultures were also transformed by a viral injection.

Hacker culture was born out of a reaction to closed doors, non-disclosure agreements, and bad technology that had to be fixed by any means necessary. Against blackboxing, phreakers built so-called “blue boxes” which allowed free phone calls. The phreakers — early hackers who were telephony geeks – could construct the blue boxes because they refused to look at only the input and output of telephone networks: they were interested in the internals. The interest in buildinig hackable devices transformed the ICT industry. To some extent, the engineering techniques against stabilisation and closure became part of mainstream technologies and aided the rapid evolution of Internet. As a result, today closure and stabilisation does not necessarily mean that a product is ready for the market, rather we can say that they often set in with the sound of their death knell. A technology too rigid will not attract enough independent innovation and fall back behind competitors. If interpreted as the signs of success before, closure and stabilisation now often foreshadow failure. In conclusion, hackable devices can help companies keep up with the speed of innovation.

But how to build a technology in a way that it fends off stabilisation and evades closure, yet it still works? Here we make an analytical distinction between the technical composition of an artefact as an engineering question and the other (social, legal, political) aspects which make up engineering as a culture. There are many specific technological components which were invented to fend off stabilisation and closure, and here we briefly present two of them which are more or less functionally equivalent.

In FLOSS[2] there is often a part of the program which is called the API[3]. Its name is quite telling: besides providing an interface for end users, like buttons rendered with gradient colours and such, the application can also expose its inner functionality to the outside world so that other programs can “call” them. In essence, APIs allow developers to design programmable programs. This is how it is possible that there are thousands of “Twitter apps” which all connect to Twitters’ servers behind the scenes, yet they can look very different and offer the end user very different functionalities and graphical user interfaces too. Even if Twitter would not develop its software all the time, the API allows a whole ecosystem of third parties to reinvent it every day. So much so that Twitter actually decided to restrict the functionality of its APIs because the third party applications started to encroach on the market share of its official website and apps. APIs are not accidents which happen to digital technologies but they have been deviced as an answer to the threat of stabilisation and it is a big job to design, implement and document them so elegantly that other developers get hooked on them.

[2] Free and Open Source Software

[3] Application Programming Interface

In OSHW[4] there are similar design features called GPIOs.[5] These are pinouts whose function is not defined by the designers, but they make sure that it is possible to access and control them from the device. It is easy to see that whenever one uses an off-the-shelf microcontroller in a hardware design, usually some legs will go unused simply because the chip has a fixed amount of them. In this case the unused legs can be connected to free pins physically, and driver support provided in the software logically. The majority of hardware hacks documented on pages like Hack A Day make use of GPIOs. The open source Arduino prototyping board which launched the recent wave of exorbitant interest in Do It Yourself electronics and robotics is basically nothing more than an ATmega microprocessor hooked up to a lot of GPIO pins and supported by a user friendly ecosystem of tool chains, tools and tutorials. Its most recent rival, the Raspberry Pi single board computer puts the barrier of entry even lower, since its GPIO pins are accessible through popular programming languages like Python.

In statistics and systems theory the freedom of a system is defined by the number of states it can possibly achieve. In such a context we can say that closure and stabilisation as social processes reduce the freedom of a system. In contrast, APIs and GPIOs arguably increase the freedom of the system, so that it becomes hard to see in what direction it can evolve. They ensure compatibility with an unforeseen future. This is why we call these unfinished artefacts, because they include functional features which make it impossible to really finish the design. Therefore, they open the door for relevant social groups like hackers to more easily participate in the making of the artefact and its subsequent configurations. While we only looked at the technical aspect here, both approaches (closure-centred vs. unfinished artefacts) are embedded in a multifaceted (aesthetic, moral, legal, social, political) engineering culture. These technical features only acquire the consistency of a program because they are intertwined with structurally similar movements in other fields like FLOSS and OSHW licences in law.

As we have noted before, GPIOs and APIs are just two specific examples but there are a host of other engineering techniques which seek to achieve similar effects vis-as-vis closure and stabilisation. We list some here for reference and further informal discussion:

- *loose coupling* (modularity of design)
- *forking* (universal ownership)
- *distributed revision control* (rollback of changes)
- *disassemblers* (reverse engineering)
- *live coding* (programming as a real time public performance)
- and ironically, *closures* in functional programming

Of course, what we have said so far is highly dialectic, so that it can be repeated

[4]Open Source Hardware
[5]General Purpose Input/Output

backwards too. The only reason we can describe the functional features which allow the maintainence of interpretative flexibility over the whole lifespan of an artefact is that open standards for APIs (like JSON-RPC) and GPIOS (like RS232) can be highly stable over a comparatively long time frame and held together by powerful social forces, be it the quasi-religious dedication of hackers to old and proven ways of doing things (like the text file as the universal interface), or the grip of a cartel involving powerful corporations over a segment of the market (like OpenGL).

Proving the solidity of the core ideas behind the original conceptualisation of closure and stabilisation, we can safely say that even the functional features which serve to protect technologies from these two things need to stabilise and achieve closure, if they are ever supposed to actually work. As a complement to the hyperinflated innovation ecology which hackers helped to create, there is in fact a whole set of concrete technological solutions as well as design principles, epitomised by the basic UNIX[6] tools and the so-called UNIX philosophy, that hackers helped to stabilise and to which they hold fast at least since the inception of UNIX in 1974.

So in the final analysis what matters is not to destabilise everything. The debates, discussions, and development work of hackers is about how stabilisation and closure are socially constructed in the case of each piece of technology. Hackers agree that for the sake of the social good, some technologies should be stabilised and agree that some others have to be opened, while about yet another group of technologies there are long lasting controversies. In all these cases however, the central question is how to configure closure and stabilisation through their technical design in a more efficient and democratic way? Of course, pursuing these questions hackers run into all the other problem fields like aesthetics, law, economic interests, etc.

Criticism of closure and stabilisation in SCOT

The way in which the concepts of closure and stabilisation evolved is rather ironic since after an initial rough consensus emerged around their meaning, researchers started to understand them in various, increasingly complex ways. In this sense closure and stabilisation did not become more closed or stable beyond a certain critical point – something to take into account when evaluating their universality!

Here we take up two lines of critique which were targeted at the initial formulation of closure and stabilisation.

One was developed with the benevolent participation of the original authors and epitomised in the edited volume Oudshoorn and Pinch (2003). *It focused on*

[6](Not an acronym. According to Raymond 1992, *Dennis Ritchie says that the UNIX spelling originally happened in CACM's 1974 paper The UNIX Time-Sharing System because "we had a new typesetter and troff had just been invented and we were intoxicated by being able to produce small caps."*)

the co-construction of users and technology, collecting papers which showed that closure and stabilisation do not end when the product makes it into the market. Closure is not merely another name for the end of the formal design phase but describes a specific social process which can take place around technological artefacts any time. Users have a significant role in shaping technology and their decisions about whether to adopt the technology, adapt it to their life, or refuse to be enrolled have their particular consequences for the trajectory that a certain technology carves out in the social fabric. Furthermore, each relevant user group reshapes the technology and makes it meaningful in its own social context.

Referring to a widely read article (Lindsay 2003) which foregrounds the role of users in determining the meaning and use of a technology, we can say for instance that the old-school users of the TRS-80 brought on a second wave of closure and stabilisation when they were abandoned by the designers, producers, marketers, distributors, and technical support of their vintage computers, so that they had to replay these traditionally early stages of technology research and development.

The other line of criticism came from further afield and also went further in its attack. In the best tradition of critical theory, *it has asked for reflection on the things which are missing from the SCOT account of closure and stabilisation.* There have been many candidates.

For instance, Klein and Kleinman (2002) criticise the concept of consensus as it has been used in the literature. They argue that consensus have to be seen as a rhetorical event in the context of a discourse between uneven parties. They point out that consensus often means the mere redefinition of unresolved problems. They stress the systematic assymetries between the relevant social groups' access to participate in the process. In particular they take up the argument from Williams and Edge (1996) that – despite the colourful case studies about user repurposing of consumer products – by and large the industry has a much stronger influence on shaping technology. In a similar vein Tympas (1997) points out that the direction of research and development determines which alternative technologies become viable candidates in the future long before consumers can interfere in the process. What is worse, consumers as a social group are dispersed and disorganised, which makes it hard for them to become relevant players. Finally, there are some social groups which are outright excluded from the very beginning.

Tympas (1997) distills these criticisms into the observation that the artefact-centred approach of SCOT may overlook the possible presence of broader patterns. Therefore, in effect SCOT actively assists hegemonic discourses and assymetric power relationships by rendering them invisible in its accounts. SCOT's emphasis on agency neglects the social power structures.

Politicisation of closure and stabilisation

In addition to these two perspectives, we are bringing a third development which applies the theory of closure and stabilisation to itself. Our two points is that unfinished artefacts (e.g. eminent open source technologies) include specific functional elements which fend off stabilisation and closure, and that in the case of hackers, the question how closure and stabilisation should work as a social process is problematised by this relevant social group itself as a techno-political issue (e.g. controversy).

The choice of this case can therefore include in the theoretical account the broader social conflicts and unequalities which some critics said are missing from the SCOT account, while at the same time maintaining the focus on empirical case studies of artefacts. Thus we can develop SCOT in a direction that recognises both the agency of users to influence the stabilisation process and the way in which unevenly distributed social power structures the space of possibilities. Through our case studies we can tackle both previous critiques and help to develop the theory in a new direction, thanks to hackers' increased reflectivity of socio-technical issues. The meta-theory about how how closure and stabilisation are themselves *designed* or *inscribed* onto the functional composition of technologies, and about the ways these very engineering questions can become the subjects of wide-spread controveries helps scholars to theorise research and development as a field of unfolding social conflicts.

Therefore, what we see in the open source world is the reflection and politicisation of the questions around closure and stabilisation. Political agency in the context of technological struggles often depends on strategies for using design to distribute in time and structure in space the possibilities of closure and stabilisation. In this sense we can speak about the *design* of closure and stabilisation itself. To sum up, we argue that in open source technology, broader issues are translated into practices and technical norms in order to keep artefacts open.

References

Callon, Michel. 1986. "The Sociology of an Actor-Network: The Case of the Electric Vehicle." In *Mapping the Dynamics of Science and Technology: Sociology of Science in the Real World*, ed. Michel Callon, John Law, and Arie Rip, 29–30. London: Macmillan.

Klein, Hans, and Daniel Kleinman. 2002. "The Social Construction of Technology: Structural Considerations." *Science, Technology & Human Values* 27 (2): 28–52.

Latour, Bruno. 1999. *Pandora's Hope: Essays on the Reality of Science Studies.* Cambridge, MA: Harvard University Press.

Lindsay, Christina. 2003. "From the Shadows: Users as Designers, Producers, Marketers, Distributors, and Technical Support." In *How Users Matter: The Co-Construction of Users and Technology*, ed. Nelly Oudshoorn and Trevor Pinch, 29–50. Cambridge, MA: MIT Press.

Lukács, György. 1971 [1923]. *Történelem És Osztálytudat [History and Class Consciousness].* Budapest: Magvető.

Oudshoorn, Nelly, and Trevor Pinch, ed. 2003. *How Users Matter: The Co-Construction of Users and Technology.* Cambridge, MA: MIT Press.

Raymond, Eric S. 1992. *The New Hacker's Dictionary.* Cambridge, MA: MIT Press.

Tympas, Aristotle. 1997. "Reviewed Works: Of Bicycles, Bakelites, and Bulbs: Toward a Theory of Sociotechnological Change by Wiebe Bijker." *Isis* 88 (2): 379.

Williams, Robin, and David Edge. 1996. "The Social Shaping of Technology." *Research Policy* (25): 865–899.